# UNITED STATES PATENT AND TRADEMARK OFFICE

| APPLICATION NO. | FILING DATE | FIRST NAMED INVENTOR | ATTORNEY DOCKET NO. | CONFIRMATION NO. |
|---|---|---|---|---|
| 09/662,258 | 09/14/2000 | Judith E. Schwabe | SUN-P4175 | 1082 |

7590        07/28/2004

THELEN REID & PRIEST LLP
ATTN: David B. Ritchie
P O Box 640640
San Jose, CA 95164-0640

| EXAMINER |
|---|
| GODDARD, BRIAN D |

| ART UNIT | PAPER NUMBER |
|---|---|
| 2171 | |

DATE MAILED: 07/28/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

| | Application No. | Applicant(s) |
| *Office Action Summary* | 09/662,258 | SCHWABE, JUDITH E. |
| | Examiner | Art Unit |
| | Brian Goddard | 2171 |

*-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --*

**Period for Reply**

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM
THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed
  after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
  Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any
  earned patent term adjustment. See 37 CFR 1.704(b).

**Status**

1)☒ Responsive to communication(s) filed on *07 May 2004*.

2a)☐ This action is **FINAL**.    2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is
closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

**Disposition of Claims**

4)☒ Claim(s) *3,4,10,11,17,18 and 22-35* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *3,4,10,11,17,18 and 22-35* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

**Application Papers**

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *22 September 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

**Priority under 35 U.S.C. § 119**

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All   b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage
application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

**Attachment(s)**

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date *14,15,16 & 19*.

4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: _____ .

## DETAILED ACTION

1.     This communication is responsive to the Request for Continued Examination and

Amendment C, filed 07 May 2004.

2.     Claims 3-4, 10-11, 17-18 and 22-35 are pending in this application.  Claims 3, 10,

17, 22, 24, 27, 30 and 33 are independent claims.  In Amendment C, claims 1-2, 5-9,

12-16 & 19-21 were cancelled; claims 22-35 were added; and claims 3, 10 and 17 were

amended.  This action is non-final.

### *Claim Rejections - 35 USC § 103*

The text of those sections of Title 35, U.S. Code not included in this action can

be found in a prior Office action.

3.     Claims 3-4, 10-11, 17-18, 22-25, 27-28, 30-31 and 33-34 are rejected under 35

U.S.C. 103(a) as being unpatentable over U.S. Patent No. 5,408,665 to Fitzgerald in

view of U.S. Patent No. 6,526,571 to Aizikowitz et al. and U.S. Patent No. 5,907,704 to

Gudmundson et al.

Referring to claim 3, Fitzgerald teaches a system and method for listing public

elements in a library as claimed.  See Figures 3-4 and the corresponding portions of

Fitzgerald's specification for this disclosure.  Refer also to claims 1 and 6 for more

details of this disclosure.  In particular, Fitzgerald teaches a method for representing an

application programming interface (API) definition for a programming language library

[260], said method comprising:

creating [Librarian 265 creates] a public list [Standard Dictionary 360 (also 430):

'a list of the library's public symbols and module names' (Column 8, lines 51-59)]

including all public elements [library object modules (See Fig. 3B)] defined in said

programming language library, said public list including a class sublist [Dependency List

445] for each of said public elements, each said class sublist including all direct and

indirect public superclasses of a class ['each module it needs' (Column 11, lines 17-25)

See also Column 3, lines 13-25]; and

storing [stored in Library File 410 (See Fig. 4A)] said list.

Fitzgerald does not explicitly state that the library (260) is an object-oriented

library as claimed, and thus does not expressly state that the public elements are public

classes and interfaces. However, Fitzgerald does state that in the preferred

embodiment, the programming language specific to the system is Borland C++. See

column 5, lines 46-59 for this disclosure. C++ being an object-oriented language, this

provides direct suggestion for using an object-oriented library for Fitzgerald's library as

claimed. Furthermore, one can infer that Fitzgerald's library is object-oriented because

it stores objects. See Figures 3B-4A and the corresponding portions of Fitzgerald's

specification for this disclosure.

Aizikowitz teaches a system and method similar to that of Fitzgerald, wherein a

class dependency hierarchy is generated from an object oriented library. See Figures 1

& 2 and the corresponding portions of Aizikowitz' specification for this disclosure. In

particular, Aizikowitz teaches the practice of creating a class hierarchy (CHG) for

classes and interfaces of a Java package (object-oriented library). Furthermore,

Aizikowitz' public elements (as applied to Fitzgerald) comprise classes and interfaces as

claimed. See Figure 2; column 2, lines 58-59; and column 3, lines 40-46 of Aizikowitz'

specification for this disclosure. Finally, Aizikowitz' public hierarchically-related

elements (as applied to Fitzgerald) comprise public superclasses and public

superinterfaces of said classes and said interfaces as claimed. See Figure 2 and

column 7, lines 25-30 of Aizikowitz' specification, in light Fitzgerald's disclosure of the

Dependency List in the combination above, for this disclosure.

It would have been obvious to one of ordinary skill in the art at the time the

invention was made to apply Fitzgerald's method of creating a list of public elements

reflecting their dependencies to the object-oriented library of Aizikowitz in order to

derive the object-oriented library's dependency hierarchy in a list structure as claimed.

One would have been motivated to do so because of the suggestions provided by

Fitzgerald as above.

Neither Fitzgerald nor Aizikowitz explicitly state that each class sublist excludes

private classes as claimed. However, Aizikowitz does mention the usage of

encapsulation in conjunction with Figure 2. Gudmundson, a system and method similar

to those of Fitzgerald and Aizikowitz, shows that the purpose of encapsulation in object-

oriented programming is to hide private objects from all but the class to which they

directly belong. See e.g. the Background of the Invention section of Gudmundson's

specification for this disclosure. Thus, by Aizikowitz' disclosure of encapsulation, it

would have been obvious to one of ordinary skill in the art at the time the invention was

made that Aizikowitz' method (as combined with Fitzgerald) would display the class

sublist including the public superclasses, but excluding the private classes as claimed, given Gudmundson's disclosure of the purpose for encapsulation. One would have been motivated to combine the references as such in order to fill the void created by Aizikowitz' lack of description on encapsulation.

Referring to claim 4, the system and method of Fitzgerald in view of Aizikowitz and Gudmundson as applied to claim 1 above discloses the invention as claimed. Aizikowitz' object-oriented library (as applied to Fitzgerald) is a Java package as claimed. See Figure 1 and column 2, line 50 et seq. of Aizikowitz' specification for this disclosure.

Claims 10-11 are rejected on the same basis as claims 3-4 respectively. See the discussions regarding claims 3-4 above for the details of this disclosure.

Claims 17-18 are rejected on the same basis as claims 3-4 respectively. See the discussions regarding claims 3-4 above for the details of this disclosure.

Claims 22-23 are rejected on the same basis as claims 3-4 respectively. See the discussions regarding claims 3-4 above for the details of this disclosure.

Referring to claim 24, the system and method of Fitzgerald in view of Aizikowitz and Gudmundson as applied to claim 3 above discloses the invention as claimed. See Figure 6 and the corresponding portion of Fitzgerald's specification for this disclosure. In particular, Fitzgerald (as modified by Aizikowitz and Gudmundson) teaches a method for determining a program hierarchy, said method comprising:

receiving [Step 601] an application programming interface (API) definition file

[Standard and Extended Dictionaries] for an object-oriented library, said API definition

file including...[See the discussion regarding claim 3 above]; and

traversing the program hierarchy through the dependency list [See Fig. 6C].

Fitzgerald (as modified by Aizikowitz and Gudmundson) does not explicitly teach

the step of "indicating a first public element is a direct parent of a second public

element" as claimed.  However, looking at the structure of Fitzgerald's (as modified by

Aizikowitz and Gudmundson) Extended Dictionary described above with regard to

claims 1 and 2, one can infer that the direct parent of a specific module (public element)

is represented in the sublist (dependency list) of that module, but is not represented in

the sublist of any other modules listed in that module's sublist.  In other words, in order

to traverse Fitzgerald's (as modified by Aizikowitz and Gudmundson) hierarchy, a first

module's direct parent can be found by searching that first module's sublist to find the

second module that is not listed in the sublist for any other module in the first module's

sublist.

It would have been obvious to one of ordinary skill in the art at the time the

invention was made to program Fitzgerald's (as modified by Aizikowitz and

Gudmundson) system to traverse the Extended Dictionary's hierarchy to find a first

module's direct parent by searching that first module's sublist to find the second module

that is not listed in the sublist for any other module in the first module's sublist as

claimed.  One would have been motivated to do so because this method is easily

inferred from the structure of the Extended Dictionary, and seems to be the only method

for traversing the hierarchy possible.

Claim 25 is rejected on the same basis as claim 4 above, in light of the basis for

claim 5. See the discussions regarding claims 3, 4 and 24 above for the details of this

disclosure.

Claims 27-28 are rejected on the same basis as claims 24-25 respectively. See

the discussions regarding claims 24-25 above for the details of this disclosure.

Claims 30-31 are rejected on the same basis as claims 24-25 respectively. See

the discussions regarding claims 24-25 above for the details of this disclosure.

Claims 33-34 are rejected on the same basis as claims 24-25 respectively. See

the discussions regarding claims 24-25 above for the details of this disclosure.


4.      Claims 26, 29, 32 and 35 are rejected under 35 U.S.C. 103(a) as being

unpatentable over Fitzgerald in view of Aizikowitz and Gudmundson as applied to

claims 24, 27, 30 and 33 above, and further in view of U.S. Patent No. 5,974,255 to

Gossain et al.

Referring to claim 26, the system and method of Fitzgerald in view of Aizikowitz

and Gudmundson as applied to claim 24 above does not explicitly disclose the steps of

comparing two reconstructed program hierarchies and indicating an error when they are

inconsistent as claimed. However, Aizikowitz does disclose the need to maintain

integrity of the program hierarchy in order to maintain the signed and sealed status of

the package. See the Background and Summary of the Invention sections of Aizikowitz'

specification for this disclosure. This provides suggestion for examining the hierarchy of an API with an expected hierarchy to maintain consistency for the signed and sealed status.

Gossain discloses a method for testing the inheritance hierarchy of an object-oriented class structure by comparing the active hierarchy to a test hierarchy stored within the system. See the Figure and the Detailed Description of the Drawing section for this disclosure. Refer specifically to column 3, lines 6-14. Gossain teaches the two claimed steps as follows:

Comparing [step 18] a first program hierarchy [hierarchy of class under test (11)] with a second program hierarchy [test class hierarchy (12)]; and

Indicating an error [Column 3, lines 9-10] when said first program hierarchy is inconsistent ['when a difference between the current state and expected state...is detected' (Column 3, lines 7-8)] with said second program hierarchy.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate Gossain's method for testing class hierarchies into Fitzgerald's (as modified by Aizikowitz and Gudmundson) system such that the system would compare the hierarchy reconstructed from an Extended Dictionary for one library with the hierarchy reconstructed from a test Extended Dictionary, and indicate an error when the two hierarchies were inconsistent. One would have been motivated to do so because of Aizikowitz' suggestion described above.

Claims 29, 32 and 35 are each rejected on the same basis as claim 26 above. See the discussion regarding claim 26 for the details of this disclosure.

5.      Claims 3-4, 10-11, 17-18 and 22-23 are rejected under 35 U.S.C. 103(a) as

being unpatentable over U.S. Patent No. 6,230,314 to Sweeney et al. in view of

Aizikowitz and Gudmundson.

Referring to claim 1, Sweeney discloses a system and method for generating an

object-oriented program inheritance listing.  See Figures 3, 4 & 7 and the corresponding

portions of Sweeney's specification for this disclosure.  In particular, Sweeney teaches a

method for representing an application programming interface (API) definition for an

object-oriented program, said method comprising:

creating [Steps 703-707] a public list [Class Hierarchy (See Fig. 3 and column 3,

line 59 – column 4, line 4)] including all public classes and interfaces [set of classes]

defined in said object-oriented program, said public list including a class sublist for each

of said public classes ['for every class' (column 4, line 2)], each said class sublist

including all direct and indirect public superclasses ['the set of base classes it inherits

from is specified' (column 4, lines 2-3)]; and

storing said list [See column 19, lines 56-62].

Sweeney does not explicitly disclose that the object-oriented program used for

generating the API definition is an object-oriented library as claimed.  However,

Sweeney does disclose the use and importance of object-oriented libraries in the

background of the invention section (See column 1, lines 11-24).  This provides

suggestion for applying Sweeney's method to an object-oriented library.

Aizikowitz teaches a system and method similar to that of Sweeney, wherein a class dependency hierarchy is generated from an object oriented library. See Figures 1 & 2 and the corresponding portions of Aizikowitz' specification for this disclosure. In particular, Aizikowitz teaches the practice of creating a class hierarchy (CHG) for classes and interfaces of a Java package (object-oriented library). Furthermore, Aizikowitz' public elements (as applied to Sweeney) comprise classes and interfaces as claimed. See Figure 2; column 2, lines 58-59; and column 3, lines 40-46 of Aizikowitz' specification for this disclosure. Finally, Aizikowitz' public hierarchically-related elements (as applied to Sweeney) comprise public superclasses and public superinterfaces of said classes and said interfaces as claimed. See Figure 2 and column 7, lines 25-30 of Aizikowitz' specification, in light Sweeney's disclosure of the Dependency List in the combination above, for this disclosure.

It would have been obvious to one of ordinary skill in the art at the time the invention was made to apply Sweeney's method of creating a list of public elements reflecting their dependencies to the object-oriented library of Aizikowitz in order to derive the object-oriented library's dependency hierarchy in a list structure as claimed. One would have been motivated to do so because of the suggestions provided by Sweeney as above. It would have been further obvious to one of ordinary skill in the art at the time the invention was made to exclude private classes from the sublist as taught by Gudmundson, for the same reasons as provided above.

Referring to claim 4, the system and method of Sweeney in view of Aizikowitz as applied to claim 1 above discloses the invention as claimed. Aizikowitz' object-oriented

library (as applied to Sweeney) is a Java package as claimed. See Figure 1 and column 2, line 50 et seq. of Aizikowitz' specification for this disclosure.

Claims 10-11 are rejected on the same basis as claims 3-4 respectively. See the discussions regarding claims 3-4 above for the details of this disclosure.

Claims 17-18 are rejected on the same basis as claims 3-4 respectively. See the discussions regarding claims 3-4 above for the details of this disclosure.

Claims 22-23 are rejected on the same basis as claims 3-4 respectively. See the discussions regarding claims 3-4 above for the details of this disclosure.

### Duplicate Claims - Warning

6.      Applicant is advised that should claims 3-4 be found allowable, claims 22-23 will be objected to under 37 CFR 1.75 as being a substantial duplicate thereof. Applicant is further advised that should claims 24-26 be found allowable, claims 33-35 will be objected to under 37 CFR 1.75 as being a substantial duplicate thereof. When two claims in an application are duplicates or else are so close in content that they both cover the same thing, despite a slight difference in wording, it is proper after allowing one claim to object to the other as being a substantial duplicate of the allowed claim. See MPEP § 706.03(k).

### Response to Arguments

7.      Applicant's arguments with respect to claims 3, 10 and 17 have been considered but are moot in view of the new ground(s) of rejection.

Referring to applicant's remarks on pages 12-13 and 15-16 regarding the 35

U.S.C § 103 rejections: Applicant argued that neither Fitzgerald, Aizikowitz nor

Sweeney, taken alone or in combination, suggests or discloses creating a public

list...where the public list includes a class sublist for each of the public classes, and

where each class sublist includes all direct and indirect public superclasses of a class

*and excludes private classes.*

The examiner disagrees for the following reasons: Aizikowitz (and therefore both

combinations) discloses the usage of encapsulation from an object-oriented

programming methodology standpoint. As was well established at the time of

applicant's invention (See Gudmundson in the combination above), part of

encapsulation is the hiding of private classes/objects/elements from all but that to which

they directly belong. In other words, a private class, object or element (data) can only

be viewed and manipulated by its encapsulated class. Thus, Aizikowitz directly

suggests that the class sublist excludes private classes as claimed, through the

disclosure of encapsulation. The combination of prior art above therefore discloses

each and every element of applicant's claimed invention.


## Conclusion

8.      Any inquiry concerning this communication or earlier communications from the

examiner should be directed to Brian Goddard whose telephone number is 703-305-

7821. The examiner can normally be reached on M-F, 9 AM - 5 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, Safet Metjahic can be reached on 703-308-1436. The fax phone number for

the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the

Patent Application Information Retrieval (PAIR) system. Status information for

published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only.

For more information about the PAIR system, see http://pair-direct.uspto.gov. Should

you have questions on access to the Private PAIR system, contact the Electronic

Business Center (EBC) at 866-217-9197 (toll-free).

bdg
19 July 2004

FRANTZ COBY
PRIMARY EXAMINER